Linear Quadratic Path-Following via Online Trajectory Speed Optimization

Duarte J. Antunes, *Senior Member IEEE*

***Abstract*—Consider a linear system subject to stochastic disturbances and a path to be followed by a system's output. The path-following problem is posed here as choosing both the control input and the speed along the path to minimize the expected value of a quadratic function of the control input and of the error between the output and the resulting trajectory. The optimal control input policy for the deterministic version (no stochastic disturbances) is first provided and shown to be the sum of linear state feedback and path-dependent components, as for the twin linear quadratic trajectory-tracking problem. This policy is proven to also be optimal for the original stochastic problem when the path is a straight line. For general paths, it acts as a certainty equivalent policy that is shown to improve the cost of the optimal trajectory-tracking policy for any given trajectory, both when it can be exactly computed and when proposed approximate methods are used otherwise.**

***Index Terms*—Linear Quadratic Control, Path Following, Optimal Control, Stochastic Control**

I. INTRODUCTION

In many control applications, rather than having a system's output track a time-parameterized reference, it is more important to minimize the distance of the output to a given path while meeting soft time constraints, such as the desired speed. These applications include machine tooling (e.g., CNC, laser profiling) [1]–[3], robotic manipulators [4], [5], and autonomous vehicles [6], [7]. Path-following addresses these problems, where precise trajectory-tracking in a timewise sense is sacrificed for output accuracy. It typically leads to time-invariant motion control policies, consistently showing superior results to trajectory tracking in the mentioned applications, requiring less demanding actuation [8] and avoiding corner cutting [9]. The price to pay is challenging nonlinear and non-convex optimization problems [2], [10]–[12], even for linear systems, with few results formalizing improvements with respect to trajectory tracking [10], [11].

A popular formulation in the autonomous vehicle community, refers to path-following as an online control law addressing two tasks [8]: (i) reach the desired path as a function of a so-called path variable, and (ii) satisfy an additional dynamic specification, often through controlling the path variable. A prime example of the path variable is the longitudinal position of the vehicle along the path obtained through a numerical projection [6], [7], [13]. Similarly in spirit, but geared towards machine tooling applications, some approaches aim at reducing the countering error, which is the output deviation from the desired contouring path [2]. These approaches include cross-coupled control [1] and model predictive control (MPC) [2], [3]. Another problem formulation, popular in the machine tooling and robotic manipulation areas, is motivated by a suboptimal two-step approach for handling time-optimal motion planning problems [4]: a geometric path is first computed to meet task-specific requirements (e.g., obstacle avoidance) and then a time-optimal trajectory along the geometric path is computed, taking into account state and input constraints. It has been tackled with several approaches, such as dynamic programming [5], and several assumptions, some of which remarkably lead to convex problems [4]. While path-following in this sense refers to an off-line planning problem, online control methods compatible with this off-line planning have also been proposed [14]. Additional ideas are the online trajectory time scaling [15] and the choice of trajectory speed online [12], [16]. While these ideas have mainly been used in the context of time-optimal problems, an error optimization approach is followed in the present paper. Only a few articles followed this approach, see [17] considering nonlinear robotic systems and [12] considering MPC, but derived very different results. Also related, [10], [11] consider an error performance index and use (instead of optimizing) the path speed freedom to show that the limitations of trajectory tracking for non-minimum phase systems are not present in path-following. In all these articles, process disturbances are not considered; here, stochastic disturbances are considered and a set of new results is provided.

The present paper considers a linear system subject to stochastic disturbances and a cost penalizing the error between an output of the system and a trajectory along the path whose speed can be chosen online. The trajectory's speed and control input policies aim to minimize the expected value of a quadratic function of this error and of the control input. The paper provides three main contributions. Considering first the case where disturbances are not present, it shows that the optimal policy for the control inputs can be decomposed into linear state feedback and path-dependent components, as for the twin linear quadratic trajectory tracking problem. Second, it shows that this policy is also optimal when stochastic disturbances are present, provided that the path is a straight line, i.e., certainty equivalence [18] holds; here, the expected quadratic cost can be exactly computed based on Riccati equations. Third, it proposes to use this policy as a certainty equivalent policy [18] when the path is non-linear and stochastic disturbances are present and provides an algorithm building upon gradient search and MPC that guarantees an improved cost with respect to the optimal trajectory tracking policy. The crucial enabling fact for these three results is that the cost of the optimal trajectory tracking policy admits a closed-form expression under the assumptions of linearity and no input and state constraints. These strong properties are not guaranteed by more powerful and general-purpose methods that do not need these assumptions (e.g., MPC). This and other facts are

The author is with the Control Systems Technology Group, Dep. of Mechanical Eng., Eindhoven University of Technology, the Netherlands. D.Antunes@tue.nl.

illustrated through numerical examples.

The remainder of the paper is organized as follows. The optimal path-following problem with trajectory speed optimization is formulated in Section II. The main results and algorithms are given in Section III. A numerical example is provided in Section IV. Section V presents concluding remarks. The proofs of the results are given in the appendix.

II. Problem formulation and Preliminaries

Consider a discrete-time system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ z_k &= C_1 x_k \end{aligned} \tag{1}$$

where, $x_k \in \mathbb{R}^n$ is the state, $z_k \in \mathbb{R}^p$ is the output, and $u_k \in \mathbb{R}^m$ is the input at time $k \in \mathsf{H} := \{0, 1, \ldots, h-1\}$, over a finite-horizon $h \in \mathbb{N}$. The disturbance inputs w_k are assumed to be zero-mean independent and identically distributed random variables with covariance $W = \mathbb{E}[w_k w_k^\top]$ for every k. A spatial path $\gamma(s) \in \mathbb{R}^p$ parameterized by $s \in (a, b)$ with $a \in \mathbb{R} \cup \{-\infty\}$ and $b \in \mathbb{R} \cup \{\infty\}$, $a < b$, is considered with continuously differentiable γ. The output z_k should follow the spatial path $\gamma(s)$ considered in the interval, $s \in [\underline{s}, \bar{s}]$, with $\underline{s} \geq a$ and $\bar{s} \leq b$, in the sense that the errors

$$\min_{s \in [\underline{s}, \bar{s}]} \|z_k - \gamma(s)\|, k \in \mathsf{H}, \qquad z_h - \gamma(\bar{s}) \tag{2}$$

are small (ideally zero). This is formulated as follows. Let

$$s_{k+1} = s_k + v_k, \qquad k \in \mathsf{H}, \tag{3}$$

with $s_0 = \underline{s}$ so that the sampled reference/trajectory takes the form $r_k = \gamma(s_k), \quad k \in \mathsf{H} \cup \{h\}$ for some control inputs v_k. These are denoted by (trajectory) speeds, although they do not necessarily correspond to physical speeds. Moreover, let $\xi_k = \begin{bmatrix} x_k^\top & s_k \end{bmatrix}^\top$. The path-following problem is posed as the following stochastic optimal control problem: find $\pi = (\pi_u, \pi_v)$ where $\pi_u = \{\mu_0, \mu_1, \ldots, \mu_{h-1}\}$ is a policy for $u_k = \mu_k(\xi_k) \in \mathbb{R}^m$ and $\pi_v = \{\sigma_0, \sigma_1, \ldots, \sigma_{h-1}\}$ is a policy for $v_k = \sigma_k(\xi_k) \in \mathcal{W}_k$ in order to solve

$$J_0(\xi_0) = \min_\pi \mathbb{E}[\sum_{k=0}^{h-1} \|z_k - \gamma(s_k)\|_Q^2 + \|\mu_k(\xi_k)\|_R^2 + \|z_h - \gamma(s_h)\|_{Q_h}^2] \tag{4}$$

for (1), (3), subject to $s_h = \bar{s}$, for positive semi-definite (tuning matrices) Q, Q_h and R, where $\|a\|_Q^2 := a^\top Q a$ for vector a. The first term of the running cost in (4) together with the freedom of controlling s_k captures the first goal in (2) whereas the terminal cost in (4) together with the constraint $s_h = \bar{s}$ captures the second goal in (2). As usual, a penalty on the control input is added for regularization. Sets $\mathcal{W}_k$ play an important role. Three important cases $\mathcal{W} = \mathcal{W}^\ell$, $\ell \in \{1, 2, 3\}$, specified next for $k \in \mathsf{H} \setminus \{h-1\}$ are:

$$\mathcal{W}_k^1 = [0, \bar{s} - s_k] \tag{PF-C}$$

constraining the path variable to be monotonically increasing, $\underline{s} \leq s_k \leq s_{k+1} \leq \bar{s}$, in the spirit of [10], [8],

$$\mathcal{W}_k^2 = [a - s_k, b - s_k], \tag{PF-NC}$$

imposing only that $\underline{s} \leq s_k \leq \bar{s}$ when $a = \underline{s}$ and $b = \bar{s}$ and imposing no constraints when $a = -\infty$ and $b = \infty$ and

$$\mathcal{W}_k^3 = \{\bar{v}_k\}. \tag{TT}$$

capturing trajectory tracking with $v_k = \bar{v}_k$ for given constants $\bar{v}_k$, $k \in \mathsf{H}$. For all the cases $\ell \in \{1, 2, 3\}$, $\mathcal{W}_{h-1}^\ell = \{\bar{s} - s_{h-1}\}$. While it is convenient to restrict v_k to a neighborhood of nominal values $v_k \in \mathcal{W}_k \cap [\bar{v}_k - \epsilon, \bar{v}_k + \epsilon]$, for some $\epsilon > 0$ to avoid large intervals $s_{k+1} - s_k$ corresponding to generated trajectories far from intended, this is not pursued here for brevity (see Remark 1 for further discussion).

Before moving to the main results, the optimal policy for the trajectory tracking problem is discussed.

A. Trajectory tracking

For each $k \in \mathsf{H} \cup \{-1\}$, it is convenient to define the future reference $\rho_k = \begin{bmatrix} r_{k+1}^\top & r_{k+2}^\top & \ldots & r_h^\top \end{bmatrix}^\top \in \mathbb{R}^{n_k p}$, with $n_k = h - k$. In the case of trajectory tracking, i.e., when $v_k = \bar{v}_k$ for fixed $\bar{v}_k$, $k \in \mathsf{H}$, it is given by

$$\begin{aligned} \rho_k = q_k(\bar{\nu}_k, s_k), \quad q_k(\bar{\nu}_k, s_k) = \bar{\gamma}_{n_k}(h(\bar{\nu}_k, s_k)), \\ h(\bar{\nu}_k, s_k) = F_{n_k}\bar{\nu}_k + s_k 1_{n_k} \end{aligned} \tag{5}$$

$\bar{\nu}_k = \begin{bmatrix} \bar{v}_k & \bar{v}_{k+1} & \ldots & \bar{v}_{h-1} \end{bmatrix}^\top \in \mathbb{R}^{n_k}$, and $F_{n_k} \in \mathbb{R}^{n_k \times n_k}$, $\bar{\gamma}_{n_k}(a) : \mathbb{R}^{n_k} \to \mathbb{R}^{n_k p}$ given by

$$F_{n_k} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 1 & 1 & \ldots & 1 \end{bmatrix}, \quad \bar{\gamma}_{n_k}(a) = \begin{bmatrix} \gamma(a_1) \\ \gamma(a_2) \\ \ldots \\ \gamma(a_n) \end{bmatrix}.$$

In such a case, an optimal policy π_u that minimizes (4) is

$$u_k = K_k x_k + L_k \rho_k, \quad k \in \mathsf{H}, \tag{6}$$

and the resulting optimal costs-to-go (defined similarly to (4) but with cost starting at a given k rather than 0) are

$$J_k^{\mathrm{TT}}(\xi_k, \bar{\nu}_k) = x_k^\top P_k x_k + 2x_k^\top N_k \rho_{k-1} + \rho_{k-1}^\top M_k \rho_{k-1} + d_k \tag{7}$$

where, for $k = h$,

$$P_h = C_1^\top Q_h C_1, \quad N_h = -C_1^\top Q_h, \quad M_h = Q_h, \quad d_h = 0,$$

and, for $k \in \{h-1, h-2, \ldots, 0\}$,

$$\begin{aligned} P_k &= A^\top P_{k+1} A + C_1^\top Q C_1 \\ &\qquad - A^\top P_{k+1} B (R + B^\top P_{k+1} B)^\dagger B^\top P_{k+1} A \\ K_k &= -(R + B^\top P_{k+1} B)^\dagger B^\top P_{k+1} A \\ L_k &= -(R + B^\top P_{k+1} B)^\dagger B^\top N_{k+1} \\ N_k &= \begin{bmatrix} -C_1^\top Q_h & (A + BK_k)^\top N_{k+1} \end{bmatrix} \\ M_k &= \begin{bmatrix} Q & 0 \\ 0 & M_{k+1} - N_{k+1}^\top B (R + B^\top P_{k+1} B)^\dagger B^\top N_{k+1} \end{bmatrix} \\ d_k &= \sum_{\ell=k+1}^{h} \mathrm{tr}(P_\ell W) \end{aligned}$$

with $(\cdot)^\dagger$ and $\mathrm{tr}(\cdot)$ denoting the pseudo-inverse and the trace.

This fact can be obtained by combining standard results for discrete-time preview control [19] and stochastic optimal control [18, Vol 1, Ch. 3] and relies on the independence of

the w_k. Policy (6) is optimal both when $w_k = 0$ for every $k \in \mathbb{N}_0 := \mathbb{N} \cup \{0\}$ ($W = 0$) and when stochastic disturbances are considered; hence certainty equivalence holds for this twin trajectory tracking problem (according to [18, Vol I, Sec 1.3], certainty equivalence holds if '*the optimal policy is unaffected when the disturbances are replaced by their means*').

III. Main methods and results

Section III-A provides the optimal policy for a deterministic version of the path-following problem (4). Section III-B shows that this policy is also optimal for the stochastic version when the path is a straight line. In Section III-C, the performance of this policy and of a proposed approximate methods is shown to improve that of any given trajectory-tracking policy.

A. Optimal policy for deterministic version

The first result considers $w_k = 0$ for every $k \in \mathsf{H}$, in which case the expected value in (4) can be removed. For a vector $a \in \mathbb{R}^n$, $a \geq 0$ ($a \leq 0$) indicates that all the components are non-negative (non-positive), i.e., $a_i \geq 0$ ($a_i \leq 0$) for every i; I_n, 1_m, $0_{m\times n}$, 0_m denote the $n \times n$ identity matrix, the row vector of n ones, the $m \times n$ matrix with zero entries respectively, and $0_m = 0_{m\times m}$. Dimensions are often omitted.

Theorem 1. Suppose that $w_k = 0$ for every $k \in \mathsf{H}$ for system (1). Then, an optimal policy for (4) is

$$\begin{aligned} u_k &= K_k x_k + L_k \rho_k^*(\xi_k), \quad k \in \mathsf{H}, &(8)\\ v_k &= \begin{bmatrix} 1 & 0_{1\times(n_k-1)} \end{bmatrix} \nu_k^*(\xi_k) &(9)\end{aligned}$$

where $\rho_k^*(\xi_k) = q_k(\nu_k^*(\xi_k), s_k)$ is the future reference resulting from the following speeds

$$\begin{aligned} \nu_k^*(\xi_k) &\in \arg\min_{\nu_k \in \mathcal{V}_k^\ell} f_k(\xi_k, \nu_k) &(10)\\ f_k(\xi_k,\nu_k) &= 2x_k^\top N_k q_k^0(\nu_k,s_k) + q_k^0(\nu_k,s_k)^\top M_k q_k^0(\nu_k,s_k)\end{aligned}$$

with $q_k^0(\nu_k, s_k) = \begin{bmatrix} \gamma(s_k)^\top & q_k(\nu_k,s_k)^\top \end{bmatrix}^\top$ and $\mathcal{V}_k^\ell$ is defined as follows for each possibility for sets $\mathcal{W}_k^\ell$, $\ell \in \{1,2,3\}$:

$$\mathcal{V}_k^1 = \{\nu \in \mathbb{R}^{n_k} | 1_{n_k}^\top \nu = \bar{s} - s_k \ , \ \nu \geq 0\} \quad \text{(PF-C)}$$

$$\mathcal{V}_k^2 = \{\nu \in \mathbb{R}^{n_k} | 1_{n_k}^\top \nu = \bar{s} - s_k, \ a 1_{n_k} \leq h(\nu, s_k) \leq b 1_{n_k}\} \quad \text{(PF-NC)}$$

$$\mathcal{V}_k^3 = \{(\bar{v}_k, \ldots, \bar{v}_{h-1}) | \bar{v}_{h-1} = \bar{s} - s_{h-1}\} \quad \text{(TT)}$$

for $k \in \mathsf{H} \setminus \{h-1\}$ and $\mathcal{V}_{h-1}^\ell = \{\bar{s} - s_{h-1}\}$, $\ell \in \{1,2,3\}$.□

The optimal policy might not be unique since, e.g., the optimization of $f_k(\xi_k, \nu_k)$ with respect to ν_k might have multiple solutions. Function f_k is such that $J_k^{\mathrm{TT}}(\xi_k, \nu_k) = x_k^\top P_k x_k + f_k(\xi_k, \nu_k)$, so that the minimization with respect to ν_k can be interpreted as minimizing the cost-to-go of trajectory-tracking with respect to the trajectory itself along the path or, equivalently, the speed inputs.

Note that the optimal policy has a similar structure to the one of trajectory-tracking (under (TT), it naturally matches (6)) consisting of the sum of linear state feedback and path-dependent components. However, the latter component now depends on the state and results from the speed optimization problem (10). Sections III-C discusses how to solve (10).

B. Straight lines

Suppose that the path is a straight line described by

$$\gamma(s) = \phi + \chi s, \quad -\infty < s < \infty \tag{11}$$

for given vectors $\phi \in \mathbb{R}^p$ and $\chi \in \mathbb{R}^p \setminus \{0\}$ and for $a = -\infty$ and $b = \infty$, and that the path variable is not constrained, i.e., (PF-NC) holds. Then, as shown in the next result, policy (8), (9) boils down to

$$\begin{aligned} u_k &= \bar{K}_k x_k + \bar{L}_k(\phi + \chi \bar{s}) &(12)\\ v_k &= \Gamma_k \begin{bmatrix} x_k^\top & r_k^\top & r_h^\top \end{bmatrix}^\top &(13)\end{aligned}$$

where $\Gamma_k = -(\Pi_2^\top \tilde{M}_k \Pi_2)^\dagger \begin{bmatrix} \Pi_2^\top \tilde{N}_k^\top & \Pi_2^\top \tilde{M}_k \Pi_1 \end{bmatrix}$, $\bar{K}_{h-1} = K_{h-1}$ and $\bar{L}_{h-1} = L_{h-1}$ and, for $k \in \{h-2, h-1, \ldots, 0\}$,

$$\bar{K}_k = \tilde{K}_k + \begin{bmatrix} 0_{m\times p} & \tilde{L}_k \end{bmatrix} \Pi_2 \Gamma_k \begin{bmatrix} I_n \\ 0_{2p\times n} \end{bmatrix}$$

$$\bar{L}_k = \begin{bmatrix} 0_{m\times p} & \tilde{L}_k \end{bmatrix} (\Pi_1 + \Pi_2 \Gamma_k \begin{bmatrix} 0_{n\times 2p} \\ I_{2p} \end{bmatrix})$$

are computed iteratively for $k \in \{h-2, h-1, \ldots, 0\}$ from the Riccati equations

$$\begin{aligned} &\bar{P}_h = P_h, \quad \bar{N}_h = N_h, \quad \bar{M}_h = M_h\\ &\bar{P}_{h-1} = P_{h-1}, \quad \bar{N}_{h-1} = N_{h-1}, \quad \bar{M}_{h-1} = M_{h-1}\\ &\tilde{P}_k = A^\top \bar{P}_{k+1} A + Q\\ &\qquad - A^\top \bar{P}_{k+1} B (R + B^\top \bar{P}_{k+1} B)^\dagger B^\top \bar{P}_{k+1} A\\ &\tilde{N}_k = \begin{bmatrix} -C_1^\top Q & (A + B\tilde{K}_k)^\top \bar{N}_{k+1} \end{bmatrix}\\ &\tilde{M}_k = \begin{bmatrix} Q & 0 \\ 0 & \bar{M}_{k+1} - \bar{N}_{k+1}^\top B(R + B^\top \bar{P}_{k+1} B)^\dagger B^\top \bar{N}_{k+1} \end{bmatrix}\\ &\tilde{K}_k = -(R + B^\top \bar{P}_{k+1} B)^\dagger B^\top \bar{P}_{k+1} A\\ &\tilde{L}_k = -(R + B^\top \bar{P}_{k+1} B)^\dagger B^\top \bar{N}_{k+1}\end{aligned}$$

$$\begin{aligned}\begin{bmatrix} \bar{P}_k & \bar{N}_k \\ \bar{N}_k^\top & \bar{M}_k \end{bmatrix} &= \begin{bmatrix} \tilde{P}_k & \tilde{N}_k \Pi_1 \\ \Pi_1^\top \tilde{N}_k^\top & \Pi_1^\top \tilde{M}_k \Pi_1 \end{bmatrix}\\ &\quad - \begin{bmatrix} \tilde{N}_k \Pi_2 \\ \Pi_1^\top \tilde{M}_k \Pi_2 \end{bmatrix} (\Pi_2^\top \tilde{M}_k \Pi_2)^\dagger \begin{bmatrix} \Pi_2^\top \tilde{N}_k^\top & \Pi_2^\top \tilde{M}_k \Pi_1 \end{bmatrix}\end{aligned}$$

As also shown next, policy (12), (13) is also optimal for the original stochastic problem, i.e., certainty equivalence holds.

Theorem 2. Policy (12), (13) is an optimal policy for problem (4) when the path is a straight line described by (11) and the path variable is unconstrained, i.e., (PF-NC) holds. In particular, (12), (13) coincide with (8), (9) when $w_k = 0$ for every $k \in \mathsf{H}$ and both provide unique values. Moreover, the resulting optimal cost $\bar{J}_0^{\mathrm{PF}}(\xi_0) = J_0(\xi_0)$ is given by

$$\bar{J}_0^{\mathrm{PF}}(\xi_0) = \begin{bmatrix} x_0^\top & | & r_0^\top & (\phi + \chi\bar{s})^\top \end{bmatrix} \begin{bmatrix} \bar{P}_0 & \bar{N}_0 \\ \bar{N}_0^\top & \bar{M}_0 \end{bmatrix} \begin{bmatrix} x_0 \\ \hline r_0 \\ (\phi + \chi\bar{s}) \end{bmatrix} + \sum_{k=1}^{h} \mathrm{tr}(\bar{P}_k W).$$

Furthermore, $\bar{J}_0^{\mathrm{PF}}(\xi_0) \leq J_0^{\mathrm{TT}}(\xi_0, \bar{\nu}_0)$, for every $\xi_0 = (x_0, s_0) \in \mathbb{R}^{n+1}$ and any given $\bar{\nu}_k = (\bar{v}_0, \ldots, \bar{v}_{h-1}) \in \mathbb{R}^h$, with $\bar{v}_{h-1} = (\bar{s} - s_{h-1})$. □

The case of a linear path yields the strongest results as one can exactly quantify the cost gain when using path-following versus trajectory-tracking as both $\bar{J}_0^{\mathrm{PF}}(\xi_0)$ and $J_0^{\mathrm{TT}}(\xi_0, \bar{\nu}_0)$ admit closed-form expressions. Note that while (4) is quadratic in the state $\xi_k = \begin{bmatrix} x_k^\top & s_k \end{bmatrix}^\top$ when the path is linear, the proof does not follow trivially from the standard linear quadratic control result [18] due to the terminal constraint $s_h = \bar{s}$.

C. Cost improvement and approximate methods

While policy (8), (9) is, in general, not optimal for (4), it can still be applied as a certainty equivalent policy. This section shows that it can only improve the cost of the optimal trajectory-tracking policy (6) for any given trajectory. The result does not need (10) to be solved exactly, but rather approximate solutions, denoted by $\tilde{\nu}_k$, $k \in \mathsf{H}$, to be found that satisfy the conditions discussed next. Let $\tilde{\nu}_k = \tilde{\psi}_k(\xi_k)$, $\tilde{\nu}_k = \begin{bmatrix} \tilde{v}_k & \tilde{v}_{k+1} & \ldots \tilde{v}_{h-1} \end{bmatrix}^\top \in \mathcal{V}_k^\ell$, for a given $\ell \in \{1, 2, 3\}$ denote the retrieved solution by the approximate method at time k and let $\lambda_k : \mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$, $\lambda_k(\tilde{\nu}_k) = \begin{bmatrix} \tilde{v}_{k+1} & \ldots \tilde{v}_{h-1} \end{bmatrix}^\top$ be maps that extract the tail of $\tilde{\nu}_k$ by removing the first component $\tilde{v}_k$. Note that $\tilde{\nu}_k$ is, in general, a function of the state ξ_k, and $\tilde{\psi}_k$ is used to express this dependency. Given a fixed trajectory characterized by $\bar{\nu}_0 \in \mathcal{V}_0^4$, and associated with a trajectory-tracking policy to be compared with path-following, the approximate method should satisfy the following two conditions:

$$f_0(\xi_0, \tilde{\nu}_0) \leq f_0(\xi_0, \bar{\nu}_0), \quad \forall \xi_0 \in \mathbb{R}^{n+1}, \tag{14}$$

$$f_k(\xi_k, \tilde{\psi}_k(\xi_k)) \leq f_k(\xi_k, \lambda_{k-1}(\tilde{\psi}_{k-1}(\xi_{k-1}))), \quad \forall \xi_{k-1} \in \mathbb{R}^{n+1}, w_{k-1} \in \mathbb{R}^n, k \in \mathsf{H}\backslash\{0\}. \tag{15}$$

Intuitively, condition (14) requires that, at $k = 0$ and for every initial state ξ_0, the approximate algorithm retrieves a trajectory that leads to a non-larger expected cost than that obtained with trajectory-tracking characterized by $\bar{v}_0$. This connection can be seen by noting that cost (7) and function f_k in (10) are related by $J_k^{\mathrm{TT}}(\xi_k, \nu_k) = x_k^\top P_k x_k + f_k(\xi_k, \nu_k) + d_k$. Moreover, condition (15) requires that, at time k, the approximate method retrieves a trajectory that leads to a non-larger cost than that associated with the tail of the trajectory computed at the previous time step $k-1$. These conditions are naturally met when (10) can be solved exactly (as in the case of straight lines, see Section III-B) and $\tilde{\nu}_k = \nu_k^*$. An approximate method that satisfies (14), (15) is provided in the sequel. The resulting approximate policy is

$$u_k = K_k x_k + L_k \tilde{\rho}_k(\xi_k), \quad k \in \mathsf{H}, \tag{16}$$

$$v_k = \begin{bmatrix} 1 & 0_{1\times(n_k-1)} \end{bmatrix} \tilde{\psi}_k(\xi_k) \tag{17}$$

where $\tilde{\rho}_k(\xi_k) = q_k(\tilde{\psi}_k(\xi_k), s_k)$ is the reference resulting from the speed profiles provided by an approximate method that meets (15). The mentioned result is stated next.

Theorem 3. Suppose that an approximate method for (10) satisfies (14), (15) for a given $\bar{\nu}_0 \in \mathcal{V}_0^3$ and for a given $\ell \in \{1, 2, 3\}$. Then the cost (4) of policy (16), (17), denoted by $\tilde{J}_0^{\mathrm{PF}}(\xi_0)$ for initial condition $\xi_0 \in \mathbb{R}^{n+1}$, satisfies

$$J_0^{\mathrm{PF}}(\xi_0) \leq J_0^{\mathrm{TT}}(\xi_0, \bar{\nu}_0), \text{ for every } \xi_0 \in \mathbb{R}^{n+1}. \qquad \square$$

One method that meets (14), (15) is the following gradient-based method, also inspired by MPC. The idea behind this method is to optimize the trajectory velocities only in a horizon $\bar{h}$ and consider velocities at times $k \in \{0, 1, \ldots, h-1-\bar{h}\}$ taking the form

$$\hat{\nu}_k = \begin{bmatrix} \underline{v}_k & \ldots & \underline{v}_{k+\bar{h}-2} & \grave{v}_{k+\bar{h}-1} & \bar{v}_{k+\bar{h}} & \ldots & \bar{v}_{h-1} \end{bmatrix}^\top \tag{18}$$

where $\underline{\nu}_k = \begin{bmatrix} \underline{v}_k & \ldots & \underline{v}_{k+\bar{h}-2} \end{bmatrix}^\top$ if a vector of free variables to be optimized, $\bar{\nu}_{k+\bar{h}} = \begin{bmatrix} \bar{v}_{k+\bar{h}} & \ldots & \bar{v}_{h-1} \end{bmatrix}^\top$ are the trajectory speed variables of the trajectory-tracking policy one wishes to improve, and $\grave{v}_{k+\bar{h}-1}$ is set to $\grave{v}_{k+\bar{h}-1} = \bar{s} - s_k - 1^\top \bar{\nu}_{k+\bar{h}} - 1^\top \underline{\nu}_k$, to ensure that the initial part of the trajectory, which is optimized, ends where the final part, which is fixed, starts. At times $k \in \{h-\bar{h}, \ldots, h-2\}$, the considered velocity profiles take the usual form $\hat{\nu}_k = \begin{bmatrix} \underline{v}_k & \ldots & \underline{v}_{h-2} & \grave{v}_{k+h-1} \end{bmatrix}^\top$ where now $\underline{\nu}_k = \begin{bmatrix} \underline{v}_k & \ldots & \underline{v}_{h-2} \end{bmatrix}^\top$ are decision variables and $\grave{v}_{k+h-1} = \bar{s} - s_k - 1^\top \underline{\nu}_k$. At time $k = h-1$, $v_{h-1} = \bar{s} - s_{h-1}$. The method assumes that a given trajectory is available that corresponds to $\bar{\nu}_0 \in \mathcal{V}_0^\ell$ and works under any assumption (PF-C), (PF-NC) corresponding to $\ell \in \{1, 2, 3\}$.

Algorithm 1: For each $k \in \mathsf{H}$, given state ξ_k, make $\hat{\nu}_0^0 = \bar{\nu}_0 \in \mathcal{V}_0^\ell$ if $k = 0$ and $\hat{\nu}_k^0 = \lambda_{k-1}(\tilde{\nu}_{k-1})$ otherwise. Then:

Iterate: $\hat{\nu}_k^{r+1} = \hat{a}_k(\xi_k, \hat{\nu}_k^r, \epsilon^r)$, for $r \in \{0, \ldots, \bar{r}-1\}$

Retrieve: $\tilde{\nu}_k = \hat{\nu}_k^{\bar{r}}$

where, for $\hat{\nu}_k$ taking the form (18) and

$$\hat{a}_k(\xi_k, \hat{\nu}_k, \epsilon) = \begin{cases} \begin{bmatrix} \underline{\nu}_k - \epsilon \nabla_{\underline{\nu}_k} f_k(\xi_k, \hat{\nu}_k) \\ \bar{s} - s_k - 1^\top \bar{\nu}_{k+\bar{h}} - 1^\top(\underline{\nu}_k - \epsilon \nabla_{\underline{\nu}_k} f_k(\xi_k, \hat{\nu}_k)) \\ \bar{\nu}_{k+\bar{h}}, \end{bmatrix} & \text{if } k \in \{0, \ldots, h-\bar{h}-1\}, \\ \begin{bmatrix} \underline{\nu}_k - \epsilon \nabla_{\underline{\nu}_k} f_k(\xi_k, \hat{\nu}_k) \\ \bar{s} - s_k - 1^\top(\underline{\nu}_k - \epsilon \nabla_{\underline{\nu}_k} f_k(\xi_k, \hat{\nu}_k)) \end{bmatrix}, & \text{if } k \in \{h-\bar{h}, \ldots, h-2\}, \\ \bar{s} - s_{h-1} & \text{if } k = h-1 \end{cases}$$

with

$$\epsilon^\ell \in \operatorname{argmin}_{\eta \in [0, \bar{\eta}_k]} f_k(\xi_k, \hat{a}_k(\xi_k, \nu_k^r, \eta))$$

$$\bar{\eta}_k = \sup_{\beta \geq 0} \{\beta | \hat{a}_k(\xi_k, \hat{\nu}_k^r, \beta) \in \mathcal{V}_k^\ell\}. \qquad \square$$

The following proposition states that in fact conditions (14), (15) hold if Algorithm 1 is used.

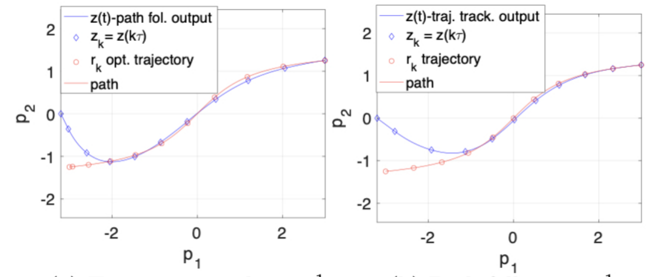
Proposition 1. The outputs $\tilde{\nu}_k$, $k \in \mathsf{H}$, of Algorithm 1 satisfy (14), (15), if $\bar{\nu}_0 \in \mathcal{V}_0^\ell$, for each $\ell \in \{1, 2, 3\}$. $\square$

Note that the approximate method in Algorithm 1 and the exact method in Theorem 1 involve searches over speeds independently of n and thus can be used for large dimensional systems. This is illustrated in the next section.

IV. Numerical examples

Consider a model consisting of two double integrators modeling the dynamics along two spatial coordinates, denoted by p_1 and p_2. This is captured by (1) with

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & A_1 \end{bmatrix}, B = \begin{bmatrix} B_1 & 0 \\ 0 & B_1 \end{bmatrix}, C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$



(a) Trajectory tracking, x_0^1 (b) Path following, x_0^1

(c) Trajectory tracking, x_0^2 (d) Path following, x_0^2

Fig. 1: Comparison between path-following and trajectory-tracking for two initial conditions $x_0^1 = \begin{bmatrix}-3.2 & 0 & 0 & 0\end{bmatrix}^\top$ and $x_0^2 = \begin{bmatrix}0 & 0 & -1 & 0\end{bmatrix}^\top$

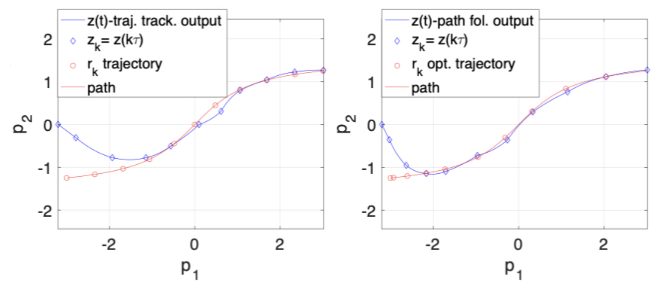


Fig. 2: Simulations with disturbances

$A_1 = \underline{A}(\tau), B_1 = \int_0^\tau \underline{A}(s)dsB_c, \underline{A}(s) = e^{A_c s}, A_c = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}$, $B_c = \begin{bmatrix}0 & 1\end{bmatrix}^\top$ and $\tau = 0.2$. For cost (4), the following matrices are considered $Q = I_2$, $Q_h = 10I_2$, $R = 0.001I_2$.

The simple path $(s, \arctan(s))$, $s \in (-3, 3)$, with $\bar{s} = 3$ is first considered. Path following is to be compared with trajectory-tracking with constant speed $v_c = (\int_{-3}^{3}(1 + (\frac{d}{ds}\arctan(s))^2)ds)/2 = 3.342$ such that the path if followed in 2 seconds. Here, s is not the arc-length, which simplifies the parameterization, but results in non-constant values v_k. Considering $h = 10$, these are computed from $s_{k+1} = s_k + v_k$, $s_0 = -3$, and $\int_{s_k}^{s_k+1}(1 + (\frac{d}{ds}\arctan(s))^2)ds = v_c$. The resulting r_k are depicted in red in Figure 1(a) and (c). First, the results with no disturbances $w_k = 0$ for every k are discussed. Figure 1(a) and (c) plot the results obtained with the fixed trajectory with constant speed v_c using trajectory-tracking for two different initial conditions $x_0^1 = \begin{bmatrix}-3.2 & 0 & 0 & 0\end{bmatrix}^\top$ and $x_0^2 = \begin{bmatrix}0 & 0 & -1 & 0\end{bmatrix}^\top$. Both the sampled output z_k and the output $z(t)$ are plotted. Figure 1(b) and (d) show the same plots but for the trajectory obtained from Algorithm 1 with $\bar{h} = h$, which approximates the solution to (10) and is run at the first iteration $k = 0$. For all numerical examples, parameter $\bar{r}$ in Algorithm 1 is set to $\bar{r} = 20$. The same values are obtained with any choice of $\mathcal{W}_k^\ell$ (and resulting $\mathcal{V}_k^\ell$), (PF-C), and (PF-NC). The plotted $z(t)$, the sampled z_k and s_k are not the actual values obtained by running the system according to Algorithm 1 since Algorithm 1 recomputes a new trajectory at the following time steps. Thus, they are rather predicted values of the state when $w_k = 0$ at time step $k = 0$. However, this shows that the predicted values have a much more sensible behavior than those obtained with trajectory-tracking since the reference is adjusted based on the state. The trajectory-tracking cost (4) with x_0^1 is 3.6694, and the predicted cost after running Algorithm 1 at $k = 0$ is 3.1819. The predicted cost at each stage $k \in \{0, 1, \ldots, 9\}$ obtained by rerunning Algorithm 1 at these stages are for $k \in \{0, 1, 2, 3\}$ $\{3.18194, 3.17381, 3.17333, 3.17329\}$ and almost identically for $k \in \{4, 5, \ldots, 9\}$ and equal to 3.17328. This illustrates the decreasing property that leads to Theorem 3 in the absence of disturbances. These predicted costs take into account the cost so far and the cost predicted with the new trajectory computed at the corresponding stage. The cost of the proposed policy coincides with the value plotted at $k = 9$, 3.17329, since in such a case Algorithm 1 has been run for all the stages.

Consider now, for the same example, white noise zero-mean Gaussian white noise disturbances in (1) with a diagonal covariance matrix W with diagonal entries $\begin{bmatrix}0 & 0.25 & 0 & 0.25\end{bmatrix}$. Figure 2 shows a sample trajectory of path-following (obtained by running Algorithm 1 for all the stages) and trajectory-tracking in the presence of these disturbances. After 100 Monte-Carlo simulations the following costs are obtained: for trajectory-tracking 3.8742 and for path-following 3.3125. The theoretical cost for trajectory-tracking (7) is 3.8814, close to the one obtained with Monte-Carlo simulations. If the trajectory would be a straight line with the same start and end points, i.e. a straight line connecting $(-3, \arctan(-3))$ and $(3, \arctan(3))$ the costs would be 3.3256 for trajectory-tracking and to 3.1241 for path-following (computed with (7)). The difference between the latter costs is not very different from the difference between the costs for the original path, showing that the cost of the approximated certainty equivalent policy is supposedly close to the cost of the optimal policy.

A more complex trajectory with intersections is considered next and it is defined as follows $\gamma(s) = \begin{bmatrix}s-3 & 0\end{bmatrix}^\top$ if $s \in [0, 3)$, $\gamma(s) = \begin{bmatrix}1-\cos(s-3) & \sin(s-3)\end{bmatrix}^\top$ if $s \in [3, 3+2\pi)$, $\gamma(s) \begin{bmatrix}s-(3+2\pi) & 0\end{bmatrix}^\top$ if $s \in [3+2\pi, 12.5]$. Algorithm 1 with horizon $\bar{h} = 4$ and $\nu_0 = v_c\begin{bmatrix}1 & 1 & \ldots & 1\end{bmatrix}^\top$, $v_c = 0.5/\tau$. is now considered with $h = 25$, and the same parameters (model and noise) as for the previous example were used. The results of path-following are shown in Figure 3. Note that the path-following algorithm is able to follow this more elaborate path with intersections successfully. The fact that a short horizon was chosen was actually beneficial in this case (see Remark 1 below). The average cost for path-following obtained with 100 Monte-Carlo simulations is 0.717 and of the trajectory-tracking policy (associated with ν_0) is 1.0776.

Let us now compare these costs with that of a standard MPC approach. Note that Algorithm 1 is inspired by MPC,

but it crucially takes into account the cost of the trajectory tracking approach after the receding horizon. This can be seen as a special terminal cost, which leads to the performance improvement property (Theorem 3). As this is the key difference, we use the same Algorithm 1, but now setting this terminal cost to zero. This amounts to considering a different f_k, which can still be written as in (10), but now with $\hat{\nu}_k = \begin{bmatrix} \underline{\nu}_k^\top & \hat{v}_{k+\bar{h}-1} \end{bmatrix}^\top \in \mathbb{R}^{\bar{h}}$ and $Q_h = 0$. This leads to a mean cost of 0.7807, still smaller than that of trajectory tracking. If we simply change R to $R = 0.01I_2$ we obtain the following costs: (i) for trajectory tracking, 6.232; (ii) for the proposed method, 5.1029 ; (iii) for MPC, 9.980. The cost of MPC is now worse than that of trajectory tracking, which highlights the key advantage of the proposed method.

In order to illustrate that the proposed policy scales well with the dimension of the system n, we consider a fifth order integrator rather than a second order integrator. This amount to changing the expression of $A_c = \begin{bmatrix} 0_{4\times 1} & I_4 \\ 0 & 0_{1\times 4} \end{bmatrix}$ and $B_c = \begin{bmatrix} 0_{1\times 4} & 1 \end{bmatrix}^\top$. To obtain reasonable trajectories the weighting matrix R has been set to $R = 0.000001I_2$. The results are shown in Figure 4 and are similar to those of the previous case shown in Figure 3. The mean run time of Algorithm 1 over the $h = 25$ times steps it is called is 0.5425 for the double integrator ($n = 4$) and 0.4503 seconds for the fifth order integrator ($n = 10$) showing that the proposed method scales well with n. It is surprising that with $n = 10$ the meantime is lower, but note that a different model and cost parameters are used. This was achieved on a MacBook laptop with no concern in optimizing the computations; a real-time implementation needs to bring the computation time below the sampling period $\tau = 0.2$, which can easily be achieved.

Remark 1. For the path with intersections depicted in Figure 3 consider a simple linear trajectory defined by $s_k = \frac{12.5-2\pi}{25}k$, $k \in \{0, 1, \ldots, 12\}$, $s_k = 2\pi + \frac{12.5-2\pi}{25}k$, $k \in \{13, 14, \ldots, 25\}$, which skips the loop and it is then equivalent to a simple straight line. The cost of trajectory-tracking for this trajectory is 0.3521, much smaller than the one obtained with path-following 0.717 (which optimized the trajectory but only in a short horizon, and thus did not find this low-cost trajectory). While better in terms of cost, the behavior is far from the intended one obtained with a short horizon $\bar{h} = 4$, beneficial in this case. As mentioned in Section II, an alternative would be to constrain the speed sets, in which case the large value $v_{12} = s_{13} - s_{12}$ would be avoided and the loop not skipped.

V. Conclusions and future work

This paper provides a framework and a set of results for path-following with stochastic disturbances that parallel those of the linear quadratic trajectory-tracking framework. Path-following is seen as a broader problem than trajectory-tracking, as it optimizes the trajectory speed online to enhance performance. Performance is measured by a quadratic cost penalizing output deviations from the path and input effort.

Some assumptions can easily be dropped. For example, to account for non-zero mean disturbances $\bar{w}_k$ it suffices to add the auxiliary state $x_{w,k+1} = x_{w,k}$ with $x_{w,0} = \mathbb{E}[\bar{w}_k]$ and set

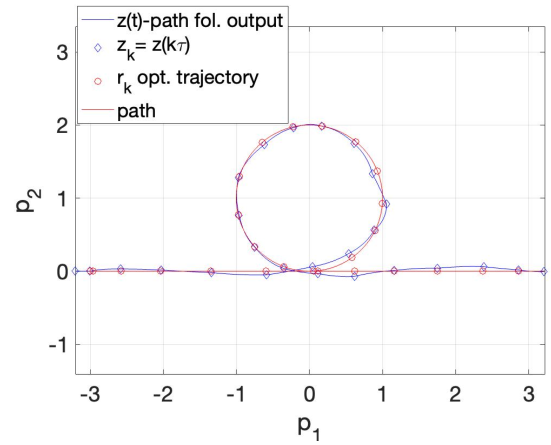


Fig. 3: Output when path-following is used for a four dimensional system and a complex trajectory with intersections

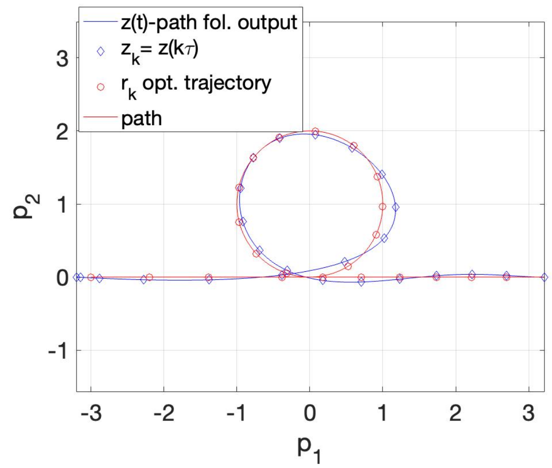


Fig. 4: Output when path-following is used for a ten dimensional system and a complex trajectory with intersections

$w_k = \bar{w}_k - x_{w,k}$. Dropping the assumptions of linearity and of no input or state constraints (e.g., to account for obstacles in the environment) is harder since no closed-form cost expression for the trajectory tracking case is known in general (which plays a crucial role in Theorems 1, 2, 3 and Proposition 1). Still, some ideas can be reused. For instance the gradient search in Algorithm 1 can be replaced by a direct search over the speeds in the horizon that rely on trajectory tracking simulations for computing costs. Determining if Theorem 3 and Proposition 1 still hold warrants further research.

Appendix

Proof of Theorem 1

The result follows from applying the dynamic programming algorithm [18, Vol. 1] to obtain the optimal policy. Let $J_h(\xi_h) = \|z_h - \gamma(s_h)\|^2_{Q_h} = x_h^\top P_h x_h + 2x_h^\top N_h \rho_{h-1} + \rho_{h-1}^\top M_h \rho_{h-1}$ with $s_h = \bar{s}$,

$$J_{h-1}(\xi_{h-1}) = \min_{u_{h-1}} \|z_{h-1} - \gamma(s_{h-1})\|^2_Q + \|u_{h-1}\|^2_R + J_h(\xi_h)$$
$$= x_{h-1}^\top P_{h-1} x_{h-1} + 2x_{h-1}^\top N_{h-1}\rho_{h-2} + \rho_{h-2}^\top M_{h-1}\rho_{h-2}$$

be the costs-to-go at stages h and $h-1$, which coincide with that of trajectory-tracking (7) since v_{h-1} and $r_h=\gamma(s_h)$ are fixed. The minimizer is $u_{h-1}=K_{h-1}x_{h-1}+L_{h-1}\gamma(s_h)$, which coincides with (8) for $k=h-1$. However, the cost-to-go at stage $h-2$ is different from that of trajectory-tracking:

$$
\begin{aligned}
J_{h-2}(\xi_{h-2}) &= \min_{v_{h-2}\in\mathcal{W}^{\ell}_{h-2}} \min_{u_{h-2}} \|z_{h-2}-\gamma(s_{h-2})\|_Q^2+\|u_{h-2}\|_R^2 \\
&\quad + J_{h-1}(\begin{bmatrix}(Ax_{h-2}+Bu_{h-2})^\top & s_{h-2}+\tau v_{h-2}\end{bmatrix}^\top) \\
&= \min_{v_{h-2}\in\mathcal{W}^{\ell}_{h-2}} x_{h-2}^\top P_{h-2}x_{h-2}+2x_{h-2}^\top N_{h-2}q^0_{h-2}(\nu_{h-2},s_{h-2}) \\
&\quad + q^0_{h-2}(\nu_{h-2},s_{h-2})^\top M_{h-2}q^0_{h-2}(\nu_{h-2},s_{h-2}) \\
&= x_{h-2}^\top P_{h-2}x_{h-2}+\min_{v_{h-2}\in\mathcal{W}^{\ell}_{h-2}} f_{h-2}(\xi_{h-2},(v_{h-2},v_{h-1})) \\
&= x_{h-2}^\top P_{h-2}x_{h-2}+\min_{\nu_{h-2}\in\mathcal{V}^{\ell}_{h-2}} f_{h-2}(\xi_{h-2},\nu_{h-2})
\end{aligned}.
$$

The minimizer that leads to the second equality (which follows from the trajectory-tracking solution as ν_{h-2} is assumed fixed while minimizing over u_{h-2}) is $u_{h-2}=K_{h-2}x_{h-2}+L_{h-2}\begin{bmatrix}\gamma(s_{h-1}) & \gamma(s_h)\end{bmatrix}^\top$ where $s_{h-1}=s_{h-2}+\tau v_{h-2}$ and v_{h-2} is the minimizer that leads to the last equality and coincides with (9). The last equality used the fact that $v_{h-1}=\frac{\bar{s}-s_{h-1}}{\tau}$ and $v_{h-2}\in\mathcal{W}^{\ell}_{h-2}$ implies $\nu_{h-2}=(v_{h-2},v_{h-1})\in\mathcal{V}^{\ell}_{h-2}$ for $\ell\in\{1,2,3\}$. In fact, if $\ell=1$, i.e., under (PF-C), $v_{h-2}\in[0,\frac{\bar{s}-s_{h-2}}{\tau}]$, which is equivalent to that $\nu_{h-2}=(v_{h-2},v_{h-1})\in\mathcal{V}_{h-2}=\{(v_{h-2},v_{h-1})|v_{h-2}\geq 0, v_{h-1}\geq 0, v_{h-2}+v_{h-1}=\frac{\bar{s}-s_{h-2}}{\tau}\}$; if $\ell=2$, $v_{h-2}\in[\frac{\underline{s}-s_{h-2}}{\tau},\frac{\bar{s}-s_{h-2}}{\tau}]$ is equivalent to $\nu_{h-2}\in\{(v_{h-2},v_{h-1})|\underline{s}\leq s_{h-2}+\tau v_{h-2}\leq\bar{s}, \underline{s}\leq s_{h-1}+\tau v_{h-1}\leq\bar{s}$ and $v_{h-2}+v_{h-1}=\frac{\bar{s}-s_{h-2}}{\tau}\}$. This fact is trivial if $\ell=3$. Assume now that

$$
\begin{aligned}
J_{k+1}(\xi_{k+1}) &= \min_{\nu_{k+1}\in\mathcal{V}^{\ell}_{k+1}} \min_{u_{k+1},\dots,u_{h-1}} \sum_{\ell=k}^{h-1}\|z_\ell-\gamma(s_\ell)\|_Q^2 \\
&\quad +\|u_\ell\|_R^2+\|z_h-\gamma(s_h)\|_{Q_h}^2 \qquad (19)\\
&= x_{k+1}^\top P_{k+1}x_{k+1}+\min_{\nu_{k+1}\in\mathcal{V}^{\ell}_{k+1}} f_{k+1}(\xi_{k+1},\nu_{k+1})
\end{aligned}
$$

and that the optimal policy is given by (8), (9) with k replaced by $k+1$. It is now established that this is also true for k, i.e.,

$$J_k(\xi_k)=\min_{v_k\in\mathcal{W}^{\ell}_k}\min_{u_k}\|z_k-\gamma(s_k)\|_Q^2+\|u_k\|_R^2+J_{k+1}(\xi_{k+1})$$

and the optimal policy at time k is given by (8), (9). Replacing (19) on the right-hand side and switching the minimization operations with respect to u_k and ν_k, leads to

$$
\begin{aligned}
&\min_{\nu_k\in\mathcal{V}^{\ell}_k}\min_{u_k,\dots,u_{h-1}}\sum_{\ell=k}^{h-1}\|z_\ell-\gamma(s_\ell)\|_Q^2+\|\mu_\ell(\xi_\ell)\|_R^2+\|z_h-\gamma(s_h)\|_{Q_h}^2 \\
&= x_k^\top P_k x_k+\min_{\nu_k\in\mathcal{V}^{\ell}_k} f_k(\xi_k,\nu_k)
\end{aligned}
$$

which, by hypothesis, is $J_k(\xi_k)$, and where the fact that $v_k\in\mathcal{W}^{\ell}_k$ and $\nu_{k+1}\in\mathcal{V}^{\ell}_{k+1}$ is equivalent to $\nu_k\in\mathcal{V}^{\ell}_k$ for every $\ell\in\{1,2,3\}$ was used. In fact, if $\ell=1$, $v_k\in[0,\frac{\bar{s}-s_k}{\tau}]$ and $\nu_{k+1}\in\{(v_{k+1},\dots,v_{h-1})|v_{k+1}\geq 0,\dots,v_{h-1}\geq 0,\sum_{r=k+1}^{h-1}v_r=\frac{\bar{s}-s_{k+1}}{\tau}\}$ is equivalent to $\nu_k\in\{(v_k,\dots,v_{h-1})|v_k\geq 0,\dots,v_{h-1}\geq 0,\sum_{r=k}^{h-1}v_r=\frac{\bar{s}-s_k}{\tau}\}$; if $\ell=2$, $v_k\in[\frac{\underline{s}-s_k}{\tau},\frac{\bar{s}-s_k}{\tau}]$ and $\nu_{k+1}\in\{(v_{k+1},\dots,v_{h-1})|\underline{s}\leq s_{r+1}+\tau v_{r+1}\leq\bar{s},\forall r\in\{k+1,\dots,h-1\},\sum_{r=k+1}^{h-1}v_k=\frac{\bar{s}-s_{k+1}}{\tau}\}$ is equivalent to $\nu_k\in\{(v_k,\dots,v_{h-1})|\underline{s}\leq s_r+\tau v_r\leq\bar{s},\forall r\in\{k,\dots,h-1\},\sum_{r=k}^{h-1}v_k=\frac{\bar{s}-s_k}{\tau}\}$. This fact is trivial if $\ell=4$. Note that in (20), in the inner optimization with respect to $u_k,\dots,u_h$, ν_k is fixed and hence the trajectory-tracking solution can be applied. The minimizer u_k is then given by (8) at time k. Moreover, the minimizer for v_k is given by (9) at time k. This concludes the proof.

Proof of Theorem 2

The result follows from applying stochastic dynamic programming [18, Vol. 1], optimizing at each step over u_k and v_k. As in the proof of Theorem 1, the costs-to-go at stages $k=h$ and $k=h-1$ and policy at time $k=h-1$ coincide with those of trajectory-tracking as $\bar{s}$ is fixed. Hereafter, the quantities for the cost-to-go in this linear case are denoted by a bar on top, i.e., $\bar{J}_h(\xi_h)=x_h^\top\bar{P}_hx_h+2x_h^\top\bar{N}_h\rho_{h-1}+\rho_{h-1}^\top\bar{M}_h\rho_{h-1}$ and $\bar{J}_{h-1}(\xi_{h-1})=x_{h-1}^\top\bar{P}_{h-1}x_{h-1}+2x_{h-1}^\top\bar{N}_{h-1}\rho_{h-2}+\rho_{h-2}^\top\bar{M}_{h-1}\rho_{h-2}+\mathrm{tr}(\bar{P}_hW)$. and $u_{h-1}=\bar{K}_{h-1}x_{h-1}+\bar{L}_{h-1}\gamma(s_h)$ where for $k=h-1$ and $k=h$ these coincide with the quantities in (7) and in (6) without the bar on top (this will not be the case for $k\in\mathsf{H}\setminus\{h-1\}$)). Note that the assertion on the policy then holds for $k=h-1$. Assume that the cost-to-go at time $k+1$ is

$$\bar{J}_{k+1}(\xi_{k+1})=\begin{bmatrix}x_{k+1}^\top & r_{k+1}^\top & r_h^\top\end{bmatrix}\begin{bmatrix}\bar{P}_{k+1} & \bar{N}_{k+1}\\ \bar{N}_{k+1}^\top & \bar{M}_{k+1}\end{bmatrix}\begin{bmatrix}x_{k+1}\\ r_{k+1}\\ r_h\end{bmatrix}+c_{k+1}.$$

where $r_{k+1}=\gamma(s_{k+1})$, $r_h=(\phi+\chi\bar{s})$ and $c_{k+1}=\sum_{\ell=k+2}^{h}\mathrm{tr}(\bar{P}_\ell W)$, and that the optimal policy at time $k+1$ is (12), (13). This holds at $k+1=h-1$. Then, the aim is to show that the cost-to-go at time k takes the same form (with $k+1$ replaced by k) and the policy at time k is (12), (13). From the stochastic dynamic programming algorithm (which implicitly uses the assumption that the w_k are independent, by considering ξ_k as the state) the cost-to-go at stage k is

$$\min_{v_k\in\mathbb{R}}\min_{u_k\in\mathbb{R}^m}\|z_k-\gamma(s_k)\|_Q^2+\|u_k\|_R^2+\mathbb{E}[\bar{J}_{k+1}(\xi_{k+1})|\xi_k].$$

Since w_k are zero-mean and letting $c_k=\mathrm{tr}(\bar{P}_{k+1}W)+c_{k+1}$, $\mathbb{E}[\bar{J}_{k+1}(\xi_{k+1})|\xi_k]=c_k+$

$$\begin{bmatrix}(Ax_k+Bu_k)^\top & r_{k+1}^\top & r_h^\top\end{bmatrix}\begin{bmatrix}\bar{P}_{k+1} & \bar{N}_{k+1}\\ \bar{N}_{k+1}^\top & \bar{M}_{k+1}\end{bmatrix}\begin{bmatrix}(Ax_k+Bu_k)\\ r_{k+1}\\ r_h\end{bmatrix}.$$

with $r_{k+1}=r_k+v_k$. Taking first the minimization with respect to u_k for fixed v_k, r_{k+1}, results in the policy and cost-to-go:

$$u_k=\tilde{K}_kx_k+\tilde{L}_k\begin{bmatrix}r_{k+1}^\top & r_h^\top\end{bmatrix} \qquad (20)$$

$$\min_{v_k\in\mathbb{R}}\begin{bmatrix}x_k^\top & \rho_{k-1}^\top\end{bmatrix}\begin{bmatrix}\tilde{P}_k & \tilde{N}_k\\ \tilde{N}_k^\top & \tilde{M}_k\end{bmatrix}\begin{bmatrix}x_k\\ \rho_{k-1}\end{bmatrix}+c_k \qquad (21)$$

where $\rho_{k-1}^\top=[r_k^\top\ r_{k+1}^\top\ r_h^\top]^\top=\Pi_1[r_k^\top\ r_h^\top]^\top+\Pi_2v_k$. Taking the minimization with respect to v_k results in (13). which replaced in the last expression and then in (13), in (20) and in (21) results in $\begin{bmatrix}x_k^\top & r_k^\top & r_h^\top\end{bmatrix}\begin{bmatrix}\bar{P}_k & \bar{N}_k\\ \bar{N}_k^\top & \bar{M}_k\end{bmatrix}\begin{bmatrix}x_k^\top & r_k^\top & r_h^\top\end{bmatrix}^\top+c_k$. which is $\bar{J}_k(x_k)$, and $u_k=\bar{K}_kx_k+\bar{L}_k\begin{bmatrix}x_k^\top & \gamma(s_h)\end{bmatrix}^\top$, as

desired. In particular, the cost of this optimal policy is $\bar{J}_0(\xi_0)$ which coincides with $\bar{J}_0^{\rm PF}(\xi_0)$. Note that this control input policy is the same for any value of W, and, for $W=0$, $c_k=0$ for every k. Then this control policy must also coincide with (8) for any $W>0$ and v_k given by (13) must coincide with the first component of ν_k^* computed by (10), when both policies provide unique values for a given state ξ_k; they still provide optimal u_k and v_k where these are not unique for a given state ξ_k, but not necessarily coincide. Performance improvement follows from Theorem 3, provided that (14), (15) hold. Recall that u_k and v_k can be computed from (8), (9). Condition (14) holds since, for any ξ_0, the search space in the minimization in (10) includes any given $\bar{v}_k$, $k\in\mathsf{H}$; (15) holds since, for any ξ_k (resulting from ξ_k-1, w_{k-1} and considered policy), the search space in the minimization in (10) at k includes the tail of the optimal trajectory computed at $k-1$.

Proof of Theorem 3

Consider a family of policies $\pi^\iota=(\pi_u^\iota,\pi_v^\iota)$, $\iota\in\{0,\dots,h\}$ where $\pi_u^\iota=\{\mu_0^\iota,\dots,\mu_{h-1}^\iota\}$, $\pi_v^\iota=\{\sigma_0^\iota,\dots,\sigma_{h-1}^\iota\}$ are policies for $u_k=\mu_k^\iota(\xi_k)$, $v_k=\sigma_k^\iota(\xi_k)\in\mathcal{W}_k^\ell$ defined as: for $\iota=0$, π^0 is the optimal trajectory-tracking policy with fixed $\bar{v}_k$, $k\in\mathsf{H}$. For $\iota\in\mathsf{H}\setminus\{0\}$, π^ι coincides with policy (16), (17), for $k\in\{0,\dots,\iota-1\}$, and, for $k\in\{\iota,\dots,h-1\}$, it is the optimal trajectory-tracking policy when the trajectory is fixed to the tail of the one computed in the previous time step $\ell=\iota-1$ ($\tilde{\nu}_\ell=\begin{bmatrix}\tilde{v}_{\ell,\iota} & \tilde{v}_{\ell,\iota+1} & \dots & \tilde{v}_{\ell,h-1}\end{bmatrix}^\top$), i.e., $v_k=\tilde{\nu}_{\iota-1,k}$, $u_k=K_kx_k+L_k\tilde{\rho}_k$, $k\in\{\iota,\iota+1,\dots,h-1\}$, with $\tilde{\rho}_k=q_k(\tilde{\nu}_k^\iota,s_k)$ and $\tilde{\nu}_k^\iota$ the tail of $\tilde{\nu}_{\iota-1}=\begin{bmatrix}\tilde{v}_{\iota-1,\iota}\dots\tilde{v}_{\iota-1,k-1} & \tilde{\nu}_k^\iota\end{bmatrix}$. Let $\tilde{J}^\iota(\xi_0)$ be the cost of π^ι. Note that $\tilde{J}^0(\xi_0)$ is equal to (7) (with $k=0$) which in turn can be written as $\tilde{J}^0(\xi_0)=x_0^\top P_0x_0+f_0(\xi_0,\bar{\nu}_0)+d_0$ Since, for $\iota=1$, the trajectory is only optimized at time $k=0$, $\tilde{J}^1(\xi_0)=x_0^\top P_0x_0+f_0(\xi_0,\tilde{\nu}_0)+d_0$. Due to (14), $\tilde{J}^1(\xi_0)\le\tilde{J}^0(\xi_0)$ for every ξ_0. Since the tail trajectory of policy ι is fixed to $\lambda_{\iota-1}(\tilde{\nu}_{\iota-1})$,

$$
\begin{aligned}
\tilde{J}^\iota(\xi_0)&=\mathbb{E}[\sum_{k=0}^{\iota-1}\|z_k-\gamma(s_k)\|_Q^2+\|\mu_k^\iota(\xi_k)\|_R^2\\
&+\mathbb{E}[\sum_{k=\iota}^{h-1}\|z_k-\gamma(s_k)\|_Q^2+\|\mu_k^{\rm TT}(\xi_k)\|_R^2+\|z_h-\gamma(s_h)\|_{Q_h}^2]\\
&=\mathbb{E}[\sum_{k=0}^{\iota-1}\|z_k-\gamma(s_k)\|_Q^2+\|\mu_k^\iota(\xi_k)\|_R^2\\
&\quad+x_\iota^\top P_\iota x_\iota+f_\iota(\xi_\iota,\lambda_{\iota-1}(\tilde{\nu}_{\iota-1}))+d_\iota]
\end{aligned}
$$

where $\mu_k^{\rm TT}(\xi_k)$ is given by (6) (corresponding to the trajectory associated with $\lambda_{\iota-1}(\tilde{\nu}_{\iota-1})$). Due to condition (15), and using the fact that $x_\iota^\top P_\iota x_\iota+f_\iota(\xi_\iota,\psi_\iota(\xi_\iota))+d_\iota=\|z_\iota-\gamma(s_\iota)\|_Q^2+\|\mu_\iota^{\iota+1}(\xi_k)\|_R^2+x_{\iota+1}^\top P_{\iota+1}x_{\iota+1}+f_{\iota+1}(\xi_{\iota+1},\lambda_\iota(\tilde{\nu}_\iota))$,

$$
\begin{aligned}
\tilde{J}^\iota(\xi_0)&\ge\mathbb{E}[\sum_{k=0}^{\iota-1}\|z_k-\gamma(s_k)\|_Q^2+\|\mu_k^\iota(\xi_k)\|_R^2\\
&\quad+x_\iota^\top P_\iota x_\iota+f_\iota(\xi_\iota,\psi_\iota(\xi_\iota))+d_\iota]\\
&=\mathbb{E}[\sum_{k=0}^{\iota}\|z_k-\gamma(s_k)\|_Q^2+\|\mu_k^{\iota+1}(\xi_k)\|_R^2+d_{\iota+1}\\
&\quad+x_{\iota+1}^\top P_{\iota+1}x_{\iota+1}+f_{\iota+1}(\xi_{\iota+1},\lambda_\iota(\tilde{\nu}_\iota))]=\tilde{J}^{\iota+1}(\xi_0).
\end{aligned}
$$

This implies that, for every ξ_0, $J_0^{\rm PF}(\xi_0)=\tilde{J}^h(\xi_0)\le\tilde{J}^{h-1}(\xi_0)\le\cdots\le\tilde{J}^1(\xi_0)\le\bar{J}^0(\xi_0)=J_0^{\rm TT}(\xi_0,\bar{\nu}_0)$.

Proof of Proposition 1

Condition (14) holds since, for any ξ_0, the starting value of Algorithm 1 is the fixed $\bar{\nu}_0\in\mathcal{V}_0^\ell$, and the gradient search steps with line search can only lead to a cost reduction of f and also ensure that $\tilde{\nu}_0\in\mathcal{V}_0^\ell$ for every choice of $\ell\in\{1,2,3\}$. Note that $\tilde{\nu}_{k-1}\in\mathcal{V}_{k-1}^\ell$ implies $\lambda_k(\tilde{\nu}_{k-1})\in\mathcal{V}_k^\ell$. Condition (15) holds since, for any ξ_{k-1}, the starting value of Algorithm 1 at time k is $\lambda_k(\tilde{\nu}_{k-1})$, $\tilde{\nu}_{k-1}=\tilde{\psi}_k(\xi_{k-1})$ which belongs to $\mathcal{V}_k^\ell$, and the gradient search steps with line search can only lead to a cost reduction of f and also ensure that $\tilde{\nu}_k\in\mathcal{V}_k^\ell$.

References

[1] Y. Koren and C. Lo, "Advanced controllers for feed drives," *CIRP Annals*, vol. 41, no. 2, pp. 689–698, 1992.

[2] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 6137–6142.

[3] P. Boucher, D. Dumur, and K. F. Rahmani, "Generalized predictive cascade control (gpcc) for machine tools drives," *CIRP Annals*, vol. 39, no. 1, pp. 357–360, 1990.

[4] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318–2327, 2009.

[5] K. Shin and N. McKay, "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Transactions on Automatic Control*, vol. 31, no. 6, pp. 491–500, 1986.

[6] M. Davidson and V. Bahl, "The scalar /spl epsiv/-controller: a spatial path tracking approach for odv, ackerman, and differentially-steered autonomous wheeled mobile robots," in *IEEE Int. Conf. on Robotics and Automation (Cat. No.01CH37164)*, vol. 1, 2001, pp. 175–180 vol.1.

[7] A. Micaelli and C. Samson, "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots," INRIA, Research Report RR-2097, 1993. [Online]. Available: https://hal.inria.fr/inria-00074575

[8] R. Skjetne, T. I. Fossen, and P. V. Kokotović, "Robust output maneuvering for a class of nonlinear systems," *Automatica*, vol. 40, no. 3, pp. 373–383, 2004.

[9] K. Moore, M. Ghosh, and Y. Chen, "Spatial-based iterative learning control for motion control applications," *Meccanica*, vol. 42, 2007.

[10] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotović, "Path-following for nonminimum phase systems removes performance limitations," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 234–239, 2005.

[11] ——, "Performance limitations in reference tracking and path following for nonlinear systems," *Automatica*, vol. 44, no. 3, Mar. 2008.

[12] T. Faulwasser, V. Hagenmeyer, and R. Findeisen, "Optimal exact path-following for constrained differentially flat systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 9875–9880, 2011, 18th IFAC WC.

[13] J. Hauser and R. Hindman, "Maneuver regulation from trajectory tracking: Feedback linearizable systems*," *IFAC Proceedings Volumes*, vol. 28, no. 14, pp. 595–600, 1995, 3rd IFAC Symposium on Nonlinear Control Systems Design 1995, Tahoe City, CA, USA, 25-28 June 1995.

[14] K. Shin and N. McKay, "Robust trajectory planning for robotic manipulators under payload uncertainties," *IEEE Transactions on Automatic Control*, vol. 32, no. 12, pp. 1044–1054, 1987.

[15] O. Dahl and L. Nielsen, "Torque-limited path following by online trajectory time scaling," *IEEE Trans. Robotics Autom.*, vol. 6, pp. 554–561, 1990.

[16] T. Faulwasser, B. Kern, and R. Findeisen, "Model predictive path-following for constrained nonlinear systems," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC)*, 2009, pp. 8642–8647.

[17] E. Szádeczky-Kardoss and B. Kiss, *On-Line Trajectory Time-Scaling to Reduce Tracking Error*. Springer Netherlands, 2009, pp. 3–14.

[18] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol I and II*, 4th ed. Athena Scientific, 2017.

[19] M. Tomizuka, "The optimal finite preview problem and its application to man-machine systems," Ph.D. dissertation, MIT, 1974.